Engenharia de Ontologias (Ontology Engineering)

Universidade Federal de Uberlândia

Faculdade de Computação

Programa de Pós-Graduação em Ciência da Computação

Prof. Fabiano Azevedo Dorça

XML Schema – Parte 2

Considerando o exemplo….

File: phone.xsd

```
<?xml version="1.0"? encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="phoneNumbers">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="title" type="xs:string"/>
          <xs:element name="entries">
            <xs:complexType><xs:sequence>
              <xs:element name="entry" minOccurs="0" maxOccurs="unbounded">
                <xs:complexType><xs:sequence>

                  <xs:element name="name">
                    <xs:complexType><xs:sequence>
                      <xs:element name="first" type="xs:string"/>
                      <xs:element name="middle" type="xs:string" minOccurs="0"/>
                       <xs:element name="last" type="xs:string"/>
                    </xs:sequence></xs:complexType>

                  </xs:element>
                  <xs:element name="phone" type="xs:string"/>
                  <xs:element name="city" type="xs:string" MinOccurs="0"/>
                </xs:sequence></xs:complexType>
              </xs:element>
            </xs:sequence></xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
  </xs:element>
</xs:schema>
```

XML Schema

Observações

- Cada elemento de um **tipo complexo** é seguido imediatamente pela **definição** desse tipo no conteúdo de seu elemento ***xs:element***.

- Por exemplo

```
<xs:element name="name">
  <xs:complexType>
    <xs:sequence>
        <xs:element name="first" type="xs:string"/>
        <xs:element name="middle" type="xs:string" minOccurs="0"/>
        <xs:element name="last" type="xs:string"/>
    </xs:sequence>
 </xs:complexType>
</xs:element>
```

XML Schema

- O tipo do elemento ***name*** é um **tipo complexo** sem um nome; É um tipo **anônimo**.
- Escrever XML Schema seguindo esta estratégia de usar tipos anônimos leva a **indentação muito profunda**.
- Tipos anônimos **não podem ser reusados**.
- Estratégia Alternativa: **Tipos Nomeados**
- Defina os **tipos complexos** na definição do XML Schema e dê a cada um deles um **nome**.
- Essas definições estarão no **nível superior** do elemento schema.

XML Schema

- O escopo de cada definição de tipo complexo **abrange todo o esquema** (a ordem não tem nenhuma conseqüência).
- **Sintaxe**: Use o sufixo "Type" ao definir um novo tipo na definição do Esquema XML.
- Exemplo:

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:complexType name="nameType">
   <xs:sequence>
      <xs:element name="first" type="xs:string"/>
      <xs:element name="middle" type="xs:string" minOccurs="0"/>
      <xs:element name="last" type="xs:string"/>
   </xs:sequence>
</xs:complexType>
<xs:complexType name="entryType">
  <xs:sequence>
      <xs:element name="name" type="nameType"/>
      <xs:element name="phone" type="xs:string"/>
      <xs:element name="city" type="xs:string" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="entriesType">
   <xs:sequence>
      <xs:element name="entry" type="entryType" minOccurs="0" maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>
<xs:complexType name="phoneType">
   <xs:sequence>
      <xs:element name="title" type="xs:string"/>
      <xs:element name="entries" type="entriesType"/>
   </xs:sequence>
</xs:complexType>
<xs:element name="phoneNumbers" type="phoneType"/>
</xs:schema>
```

XML Schema

Exemplo:

```
<xs:element name="produto" type="tipo_produto"/>

  <xs:complexType name="tipo_produto">
   <xs:simpleContent>
      <xs:extension base="xs:string">
         <xs:attribute name="prodid" type="xs:positiveInteger"/>
      </xs:extension>
   </xs:simpleContent>
</xs:complexType>
```

XML Schema

- Exemplo:

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
        xmlns:target="http://www.levijackson.net/web340/ns"
        targetNamespace="http://www.levijackson.net/web340/ns">
  <element name="assignments">
    <complexType>
      <sequence>
        <element name="assignment" type="target:assignmentInfo"
                 minOccurs="1" maxOccurs="unbounded"/>
      </sequence>
    </complexType>
  </element>
  <complexType name="assignmentInfo">
    <sequence>
      <element name="name" type="string"/>
    </sequence>
    <attribute name="id" type="string" use="required"/>
  </complexType>
</schema>
```

XML Schema

- **Exemplo**: Um elemento XML vazio com atributo:

```
<product prodid="1345"/>
```

Definição:

```
<xs:element name="product">
  <xs:complexType>
      <xs:attribute name="prodid" type="xs:positiveInteger"/>
  </xs:complexType>
</xs:element>
```

OU….

```
<xs:element name="product" type="prodtype"/>
<xs:complexType name="prodtype">
  <xs:atrribute name="prodid" type="xs:positiveInteger"/>
</xs:complexType>
```

XML Schema

- Agrupando Elementos
- Podem ser feitas referências a **blocos de código** XSD, bem como a definições de elementos.
- Especifica-se uma seção de código usando o elemento ***xs:group*** com um atributo que dá ao grupo um **nome**.
- Em posições na definição em que deseja-se utilizar o código, fornecemos um elemento ***xs:group*** com um atributo **ref** que especifica a definição do grupo.
- Exemplo:

File: products.xsd

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:group name="productGroup">
  <xs:sequence>
    <xs:choice>
        <xs:element name="productCode" type="xs:string"/>
        <xs:element name="stockNum" type="xs:string"/>
    </xs:choice>
    <xs:element name="price" type="xs:decimal"/>
  </xs:sequence>
</xs:group>
:
<xs:complexType name="exportType">
   <xs:sequence>
      <xs:element name="name" type="xs:string"/>
      <xs:group ref="productGroup"/>
   </xs:sequence>
</xs:complexType>
<xs:complexType name="localProductType">
    <xs:sequence>
      <xs:element name="name" type="xs:string"/>
      <xs:group ref="productGroup"/>
    </xs:sequence>
</xs:complexType>
:
</xs:schema>
```

XML Schema

- É possível **basear** um elemento complexo em um elemento complexo pré-existente e adicionar alguns elementos, como este:

```
<xs:element name="employee" type="fullpersoninfo"/>

<xs:complexType name="personinfo">
  <xs:sequence>
     <xs:element name="firstname" type="xs:string"/>
     <xs:element name="lastname" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```

XML Schema

```
<xs:complexType name="fullpersoninfo">
  <xs:complexContent>
    <xs:extension base="personinfo">
      <xs:sequence>
          <xs:element name="address" type="xs:string"/>
          <xs:element name="city" type="xs:string"/>
          <xs:element name="country" type="xs:string"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML Schema

- **Restrições** / Facetas (facets) XSD
  - Restrições são usadas para definir **valores aceitáveis** para elementos ou atributos XML.
  - Restrições em elementos XML são chamadas facetas.
- Restrições de Valores
  - O exemplo a seguir define um elemento "age" com uma restrição.
    - O valor da idade não pode ser inferior a 0 ou superior a 120:

XML Schema

```
<xs:element name="age">
    <xs:simpleType>
        <xs:restriction base="xs:integer">
            <xs:minInclusive value="0"/>
            <xs:maxInclusive value="120"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>
```

XML Schema

- Restrições sobre um Conjunto de Valores
- Para limitar o conteúdo de um elemento XML a um conjunto de valores aceitáveis, usaríamos a restrição de **enumeração (enumeration)**.
- O exemplo abaixo define um elemento chamado "carro" com uma restrição.
- Os únicos valores aceitáveis são: Audi, Golf, BMW:

XML Schema

```
<xs:element name="car">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:enumeration value="Audi"/>
      <xs:enumeration value="Golf"/>
      <xs:enumeration value="BMW"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

XML Schema

- Restrições a uma **série de valores**
  - Para limitar o conteúdo de um elemento XML para definir uma série de números ou letras que podem ser usados, usaríamos a restrição de **padrão**.
  - O exemplo abaixo define um elemento chamado "letter" com uma restrição.
  - O único valor aceitável é uma das letras LOWERCASE de “a” a “z”:

XML Schema

- XML Schema

```
<xs:element name="letter">
  <xs:simpleType>
    <xs:restriction base="xs:string">
        <xs:pattern value="[a-z]"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

XML Schema

- Restrição: O único valor aceitável é TRÊS das letras MAIÚSCULAS de a para z:

```
<xs:element name="iniciais">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="[A-Z][A-Z][A-Z]"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

XML Schema

- O único valor aceitável é formado por 3 das letras minúsculas ou maiúsculas de “a” a “z”:

```
<xs:element name="iniciais">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="[a-zA-Z][a-zA-Z][a-zA-Z]"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

XML Schema

- Restrição: O único valor aceitável é uma das seguintes letras: x, y, ou z:

```
<xs:element name="choice">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="[xyz]"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

XML Schema

- Restrição: O único valor aceitável é CINCO dígitos em uma seqüência, e cada dígito deve estar em um intervalo de 0 a 9:

```
<xs:element name="prodid">
  <xs:simpleType>
    <xs:restriction base="xs:integer">
      <xs:pattern value="[0-9][0-9][0-9][0-9][0-9]"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

XML Schema

- Restrição: O valor aceitável é zero ou mais ocorrências de letras minúsculas de a para z:

```
<xs:element name="letter">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="([a-z])*"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

XML Schema

- O valor aceitável é um ou mais pares de letras, sendo cada par composto de uma letra minúscula seguida de uma letra maiúscula. Por exemplo, "sToP" será validado por este padrão, mas não "Stop" ou "STOP" ou "stop":

```
<xs:element name="letter">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="([a-z][A-Z])+"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

XML Schema

- O único valor aceitável são os valores *male* ou *female* :

```
<xs:element name="gender">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="male|female"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

XML Schema

- Restrição: Deve haver exatamente oito caracteres letras minúsculas ou maiúsculas de a para z ou números de 0 a 9:

```
<xs:element name="password">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="[a-zA-Z0-9]{8}"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

XML Schema

- Restrições ao Comprimento
- Para limitar o comprimento de um valor em um elemento, usaríamos as restrições length, maxLength e minLength.
- Exemplo: O valor deve ser exatamente oito caracteres:

```
<xs:element name="password">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:length value="8"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

XML Schema

- O valor deve ter no mínimo cinco caracteres e no máximo oito caracteres:

```
<xs:element name="password">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:minLength value="5"/>
      <xs:maxLength value="8"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

XML Schema

- Tipo simples nomeado...

```
<xs:simpleType name="temperatura">
  <xs:restriction base="xs:integer">
    <xs:minInclusive value="0"/>
    <xs:maxInclusive value="100"/>
  </xs:restriction>
</xs:simpleType>
```

XML Schema

- Lista
- Exemplo

```
<xs:simpleType name="listOfDates">
  <xs:list itemType="xs:date"/>
</xs:simpleType>
```

- Permite que uma lista de datas (cada item da lista deve ser separada por espaço em branco) em seu conteúdo.

XML Schema

- Union
- Criar um tipo a partir da união de outros tipos já existentes.
- Exemplo:

XML Schema

- **Exemplo:**

```
<xs:attribute name="fontsize">
  <xs:simpleType>
    <xs:union memberTypes="fontbynumber fontbystringname" />
  </xs:simpleType>
</xs:attribute>


<xs:simpleType name="fontbynumber">
  <xs:restriction base="xs:positiveInteger">
    <xs:maxInclusive value="72"/>
  </xs:restriction>
</xs:simpleType>


<xs:simpleType name="fontbystringname">
  <xs:restriction base="xs:string">
    <xs:enumeration value="small"/>
    <xs:enumeration value="medium"/>
    <xs:enumeration value="large"/>
  </xs:restriction>
</xs:simpleType>
```

XML Schema

- **Exemplo:**

```
<xs:element name="jeans_size">
  <xs:simpleType>
    <xs:union memberTypes="sizebyno sizebystring" />
  </xs:simpleType>
</xs:element>


<xs:simpleType name="sizebyno">
  <xs:restriction base="xs:positiveInteger">
    <xs:maxInclusive value="42"/>
  </xs:restriction>
</xs:simpleType>


<xs:simpleType name="sizebystring">
  <xs:restriction base="xs:string">
    <xs:enumeration value="small"/>
    <xs:enumeration value="medium"/>
    <xs:enumeration value="large"/>
  </xs:restriction>
</xs:simpleType>
```

XML Schema

- **xs:import** - permite importar esquemas
- **xs:include** – permite incluir esquemas
- A diferença fundamental entre include e import é que deve-se usar import para se referir definições que estão em target namespaces diferentes; e deve-se usar include para se referir a definições que estão no mesmo target namespace
- Exemplo:

- CommonTypes.xsd

```
<?xml version="1.0" encoding="utf-16" ?>
<xs:schema targetNamespace="http://NamespaceTest.com/CommonTypes"
                xmlns:xs="http://www.w3.org/2001/XMLSchema"
                elementFormDefault="qualified">
        <xs:complexType name="AddressType">
                <xs:sequence>
                        <xs:element name="Line1" type="xs:string" />
                        <xs:element name="Line2" type="xs:string" />
                </xs:sequence>
        </xs:complexType>
        <xs:simpleType name="PriceType">
                <xs:restriction base="xs:decimal">
                        <xs:fractionDigits value="2" />
                </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="PaymentMethodType">
                <xs:restriction base="xs:string">
                        <xs:enumeration value="VISA" />
                        <xs:enumeration value="MasterCard" />
                        <xs:enumeration value="Cash" />
                        <xs:enumeration value="AMEX" />
                </xs:restriction>
        </xs:simpleType>
</xs:schema>
```

- CustomerTypes.xsd

```
<?xml version="1.0" encoding="utf-16" ?>
<xs:schema xmlns:cmn="http://NamespaceTest.com/CommonTypes"
        targetNamespace="http://NamespaceTest.com/CustomerTypes"
                xmlns:xs="http://www.w3.org/2001/XMLSchema"
                elementFormDefault="qualified">

        <xs:import schemaLocation="CommonTypes.xsd"
                namespace="http://NamespaceTest.com/CommonTypes" />
        <xs:complexType name="CustomerType">
                <xs:sequence>
                        <xs:element name="Name" type="xs:string" />
                        <xs:element name="DeliveryAddress" type =
"cmn:AddressType" />
                        <xs:element name="BillingAddress" type =
"cmn:AddressType" />
                </xs:sequence>
        </xs:complexType>
</xs:schema>
```

- OrderType.xsd

```
<?xml version="1.0" encoding="utf-16" ?>
<xs:schema xmlns:cmn="http://NamespaceTest.com/CommonTypes"
                    targetNamespace="http://NamespaceTest.com/OrderTypes"
                    xmlns:xs="http://www.w3.org/2001/XMLSchema"
                    elementFormDefault="qualified">

        <xs:import schemaLocation="CommonTypes.xsd"
namespace="http://NamespaceTest.com/CommonTypes" />
    <xs:complexType name="OrderType">
        <xs:sequence>
            <xs:element maxOccurs="unbounded" name="Item">
                <xs:complexType><xs:sequence>
                    <xs:element name="ProductName" type="xs:string" />
                    <xs:element name="Quantity" type="xs:int" />
                    <xs:element name="UnitPrice" type="cmn:PriceType" />
                </xs:sequence></xs:complexType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```

- Main.xsd

```
<?xml version="1.0" encoding="utf-16" ?>
<xs:schema xmlns:ord="http://NamespaceTest.com/OrderTypes"
                    xmlns:pur="http://NamespaceTest.com/Purchase"
                    xmlns:cmn="http://NamespaceTest.com/CommonTypes"
                    xmlns:cust="http://NamespaceTest.com/CustomerTypes"
                    targetNamespace="http://NamespaceTest.com/Purchase"
                    xmlns:xs="http://www.w3.org/2001/XMLSchema"
                    elementFormDefault="qualified">
        <xs:import schemaLocation="CommonTypes.xsd"
                        namespace = "http://NamespaceTest.com/CommonTypes" />
        <xs:import schemaLocation="CustomerTypes.xsd"
                        namespace="http://NamespaceTest.com/CustomerTypes" />
        <xs:import schemaLocation="OrderTypes.xsd"
                        namespace="http://NamespaceTest.com/OrderTypes" />
        <xs:element name="Purchase">
                <xs:complexType><xs:sequence>
                        <xs:element name="OrderDetail" type="ord:OrderType" />
                        <xs:element name="PaymentMethod" type =
"cmn:PaymentMethodType" />
                        <xs:element ref="pur:CustomerDetails" />
                 </xs:sequence></xs:complexType>
        </xs:element>
        <xs:element name="CustomerDetails" type="cust:CustomerType" />
</xs:schema>
```

- XML Document

```
<?xml version="1.0" ?>
<p:Purchase xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://NamespaceTest.com/Purchase    Main.xsd"
                xmlns:p="http://NamespaceTest.com/Purchase"
                xmlns:o="http://NamespaceTest.com/OrderTypes"
                xmlns:c="http://NamespaceTest.com/CustomerTypes"
                xmlns:cmn="http://NamespaceTest.com/CommonTypes">
    <p:OrderDetail> <o:Item>
                <o:ProductName>Widget</o:ProductName>
                <o:Quantity>1</o:Quantity>
                <o:UnitPrice>3.42</o:UnitPrice>
        </o:Item> </p:OrderDetail>
    <p:PaymentMethod>VISA</p:PaymentMethod>
    <p:CustomerDetails>
        <c:Name>James</c:Name>
        <c:DeliveryAddress>
            <cmn:Line1>15 Some Road</cmn:Line1>
            <cmn:Line2>SomeTown</cmn:Line2>
        </c:DeliveryAddress>
        <c:BillingAddress>
            <cmn:Line1>15 Some Road</cmn:Line1>
            <cmn:Line2>SomeTown</cmn:Line2>
        </c:BillingAddress>
    </p:CustomerDetails>
</p:Purchase>
```

XML Schema

- Fim